FULL Project Zero Description

Leveraging Java, create an application that simulates simple banking transactions
Requirements:

- The bank should allow 3 types of accounts to be created: customer, employee, and admin.
- All interaction with the user should be done through the console using the Scanner class
- (Project 0, part 1) All information should be persisted using serialization and text files. Ignore this step when you're doing project 0 part 2.
- (Project 0, part 2) All information should be persisted using JDBC and an Oracle DB.

CUSTOMER ACCOUNTS

- Customers of the bank should be able to register a new USER account with a username and password.
- Customers should then be able to apply for a BANK account(s). The account should be in a "pending" state, awaiting an employee or admin to approve the opening of the new bank account.
- Customers should ALSO be able to apply for joint bank accounts.
    - One customer may have MANY bank accounts, and one bank account may be owned by MANY customers. This is a many to many relationship….you can't avoid a junction table.
- Once the account is open (approved by an employee or admin), customers should be able to withdraw, deposit, and transfer funds between accounts (customers may have multiple accounts, to be clear).
- All basic validation should be done, such as trying to input negative amounts, overdrawing from accounts etc.

EMPLOYEE ACCOUNTS

- Employees of the bank should be able to view all of the customers' information
    - Account information
    - Account balances
    - Personal information
- Employees should be able to approve/deny open applications for accounts

ADMIN ACCOUNTS

- Bank admins should be able to view and edit all accounts
    - Approving/denying accounts
    - Withdrawing, depositing, transferring from all accounts
    - Canceling accounts

ADDITIONALLY

- In the restaurant specification, there is a “Tips on how to start” section is VERY much worth a read, because all of that information still applies to the banking application as well. So go read it.
- You should use the DAO design pattern for DB connectivity.
- Reasonable test coverage is expected using J-Unit
    - Folks, i’m not asking for 100% test coverage; but give me at least 5-10 tests to demonstrate that you are capable of using J-Unit. I don’t think this is a tall ask.
- Logging should be accomplished using Log4J
    - All transactions should be logged